

QT Programming

QT Book in the Library

BOĞAZİÇİ ÜNİVERSİTESİ KÜTÜPHANESİ

Site Haritası | İletişim
Site içi arama ARA

ENGLISH | BU ANASAYFA

- Anasayfa
- Katalog Tarama
- Elektronik Servisler**
- Deneme Veritabanları
- e-Dergiler
- e-Kitaplar
- Proxy Ayarları
- Referans Kaynakları
- Toplu Arama
- Tumitit & iThenticate
- Veritabanları
- Araştırma Olanakları
- Kütüphane Hakkında
- Kütüphane Kılavuzu
- Kullanıcı İşlemleri
- Kütüphane Sözlüğü
- Engelli Hizmetleri
- Kullanım Koşulları
- Sık Sorulan Sorular
- İstek / Öneri Formu

Veritabanları: | Hepsı | Referans | Müzik | Video | Gazete | **e-Kitap** | Deneme erişimi | Açık erişim |

A B C Ç D E F G H I J K L M N O P Q R S Ş T U V W X Y Z

BILGI VERİTABANLARI MOBİL KİLAVUZ KAMPÜS DİĞİ ERİŞİM

BILGI	VERİTABANLARI	MOBİL	KİLAVUZ	KAMPÜS DİĞİ ERİŞİM
Brill e-Book Collections				kampüs dışı erişim
CRCnetBASE				kampüs dışı erişim
Current Protocols (Wiley)				kampüs dışı erişim
Early English Books Online				kampüs dışı erişim
EBRARY				kampüs dışı erişim
Ebsco e-Books				kampüs dışı erişim
IEEE-Wiley eBooks Library				kampüs dışı erişim

İnternet Adresi: [www.bilkent.edu.tr](#)

Kullanıcı İşlemleri

- [Early English Books Online](#)
- [EBRARY](#)
- [Ebsco e-Books](#)
- [IEEE-Wiley eBooks Library](#)
- [InfoSci-Books](#)
- [Knovel](#)
- [Lecture Notes in Computer Science](#)
- [Lecture Notes in Mathematics](#)
- [Lecture Notes in Physics](#)
- [Life Sciences Book Series](#)
- [Oxford Scholarship Online](#)
- [Safari IT Books & Business Books](#)
- [ScienceDirect Ebooks & Reference Works](#)
- [Springer e-Kitapları](#)
- [World eBook Library](#)

34342 Bebek İSTANBUL Tel / Faks: (0212) 257 5016

Copyright © 2008 Boğaziçi Üniversitesi Kütüphanesi

E-kitaplar (e-books)

Entire Site

All Results for: **qt**

Books

Welcome to Safari Books Online

You are signed in to Safari Books Online, paid for and licensed by your academic or public library. You are accessing a Custom Safari Books Online library. This contains a subset of **41,180** titles from Safari Books Online's overall content.

Safari Books Online is the premier on-demand digital library providing over **41,180** technology, digital media, and business books and videos online to academic and library users. Individuals may access Safari Books Online's complete offering for a fee. Free trials are available.

If you are already a Safari Books Online subscriber and would like to access your individual paid account, click here to sign in.

Book & Video Titles

- [Game Programming Using Qt](#)
- [Rapid GUI Programming with Python and Qt: The Definitive Guide to PyQt Programming](#)
- [Designing and Implementing Test Automation Frameworks with Qt](#)
- [Qt 5 Blueprints](#)
- [Application Development with Qt Creator - Second Edition](#)

Safari IT Books

Entire Site

Help ▾ Bogazici University ▾ Personal Sign In

[Return to Search Results](#)

C++ GUI Programming with Qt 4

By: Jasmin Blanchette; Mark Summerfield
Publisher: Prentice Hall
Pub. Date: February 4, 2008
Print ISBN-10: 0-13-235416-0
Print ISBN-13: 978-0-13-235416-5
Web ISBN-10: 0-13-714397-4
Web ISBN-13: 978-0-13-714397-9
Pages in Print Edition: 752
Subscriber Rating: ★★★★★ [9 Ratings]

START READING ➔

Overview Table of Contents Extras Search This Book

Search qt

Read: C++ GUI Programming with Qt-4

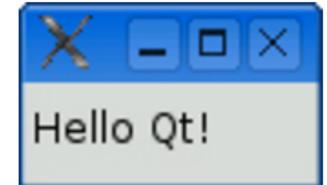
GUI Programming

- GUI programs use event driven programming model:
 - GUI programs have a lot of interactivity: there are many inputs that may come in different forms (mouse clicks, button clicks, drawing etc.)
 - A style of coding where a program's overall flow of execution is dictated by events.
 - The program loads, creates visual/non-visual objects and then waits for user input events.
 - As each event occurs, the program runs particular code to respond.
- Control flows of GUI programs have the following pattern

set-up, creation, configuration of objects (visual or non-visual)

```
do {
    event = GetNextEvent();
    process event
} while (event != quit);
```

HELLO QT Program



```
#include <QApplication>
#include <QLabel>

int main(int argc, char *argv[])
{
    int rc ;
    QApplication app(argc, argv);
    QLabel label("Hello Qt!");

    label.show();

    rc = app.exec();
    return(rc) ;
}
```

QApplication Object

- The QApplication class manages the GUI application's control flow and main settings.
- QApplication specializes GUI application with some functionality needed for widget-based applications. It handles widget specific initialization, finalization.
- For any GUI application using Qt, there is precisely one QApplication object, no matter whether the application has 0, 1, 2 or more windows at any given time. For non-widget based Qt applications, use QCoreApplication instead, as it does not depend on the widgets library.
- Widget : visual element (object)

Compiling Hello QT Program

- Put your code in a directory (there should be no other non-project related files/codes there)

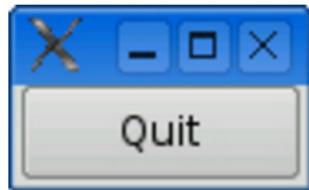
```
>qmake -project
```

```
>qmake HELLO.pro
```

————— This command will create a Makefile automatically

```
>make
```

BUTTON Example



```
#include <QApplication>
#include <QPushButton>

int main(int argc, char *argv[])
{
    QApplication app(argc, argv);

    QPushButton *button = new QPushButton("Quit");
    QObject::connect(button, SIGNAL(clicked()),
                     &app, SLOT(quit()));

    button->show();

    return app.exec();
}
```

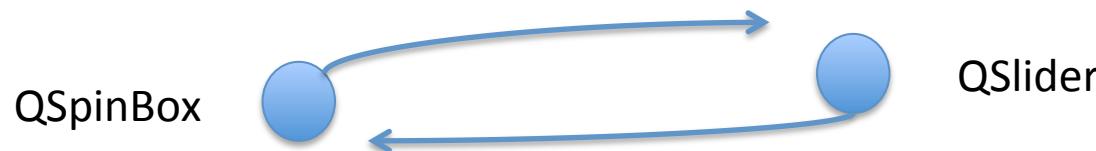
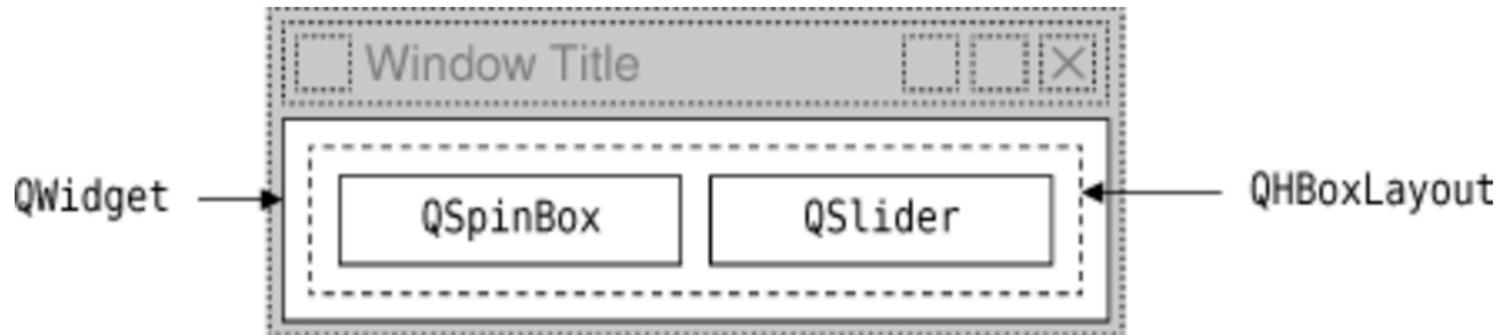
Signals and Slots

The connect() statement:

```
connect(sender, SIGNAL(signal), receiver, SLOT(slot));
```



SLIDER Example



SLIDER Example Code

```
#include <QApplication>
#include <QHBoxLayout>
#include <QSlider>
#include <QSpinBox>

int main(int argc, char *argv[])
{
    QApplication app(argc, argv);

    QWidget *window = new QWidget;
    window->setWindowTitle("Enter Your Age");

    QSpinBox *spinBox = new QSpinBox;
    QSlider *slider = new QSlider(Qt::Horizontal);
    spinBox->setRange(0, 130);
    slider->setRange(0, 130);

    QObject::connect(spinBox, SIGNAL(valueChanged(int)),
                     slider, SLOT(setValue(int)));
    QObject::connect(slider, SIGNAL(valueChanged(int)),
                     spinBox, SLOT(setValue(int)));
    spinBox->setValue(35);

    QHBoxLayout *layout = new QHBoxLayout;
    layout->addWidget(spinBox);
    layout->addWidget(slider);
    window->setLayout(layout);

    window->show();

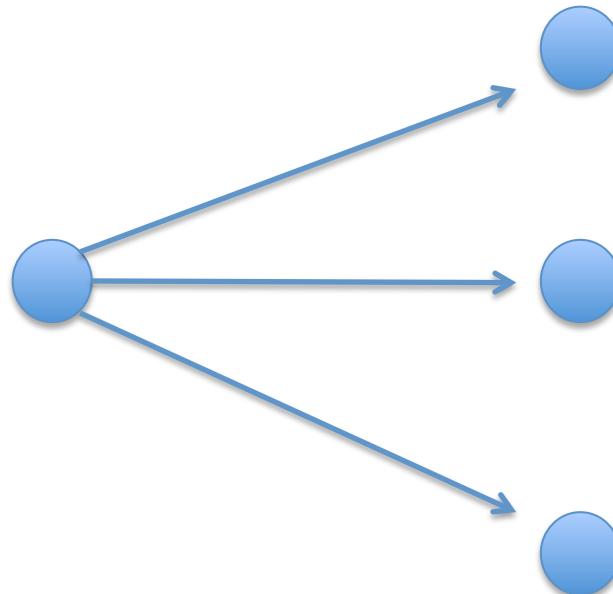
    return app.exec();
}
```

Signals and Slots

- **One signal can be connected to many slots:**

```
connect(slid... SIGNAL(valueChanged(int)),  
        spinBox, SLOT(setValue(int)));  
connect(slid... SIGNAL(valueChanged(int)),  
        this, SLOT(updateStatusBarIndicator(int)));
```

When the signal is emitted, the slots are called one after the other, in an unspecified order.

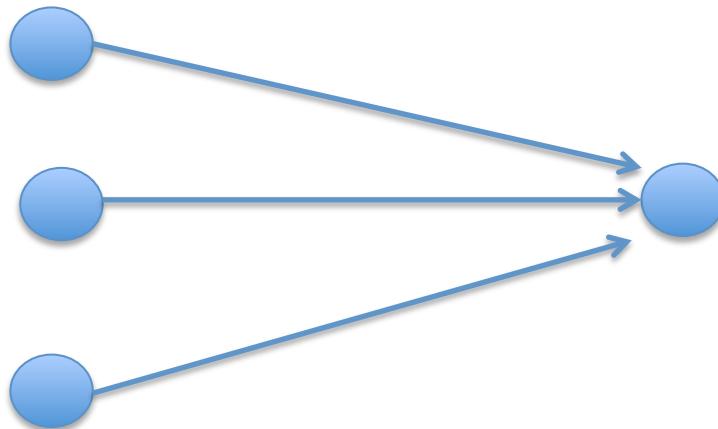


Signals and Slots

- **Many signals can be connected to the same slot:**

```
connect(lcd, SIGNAL(overflow()),  
        this, SLOT(handleMathError()));  
connect(calculator, SIGNAL(divisionByZero()),  
        this, SLOT(handleMathError()));
```

When either signal is emitted, the slot is called.



Signals and Slots

- **A signal can be connected to another signal:**

```
connect(lineEdit, SIGNAL(textChanged(const QString &)),  
        this, SIGNAL(updateRecord(const QString &)));
```

When the first signal is emitted, the second signal is emitted as well. Apart from that, signal–signal connections are indistinguishable from signal–slot connections.



Signals and Slots

- **Connections can be removed:**

```
disconnect(lcd, SIGNAL(overflow()),  
          this, SLOT(handleMathError()));
```

This is rarely needed, because Qt automatically removes all connection involving an object when that object is deleted.

Inheritance Tree for the QT Classes seen so far (partial)

